Advanced Systems Security: Assurance

Trent Jaeger
Systems and Internet Infrastructure Security (SIIS) Lab
Computer Science and Engineering Department
Pennsylvania State University

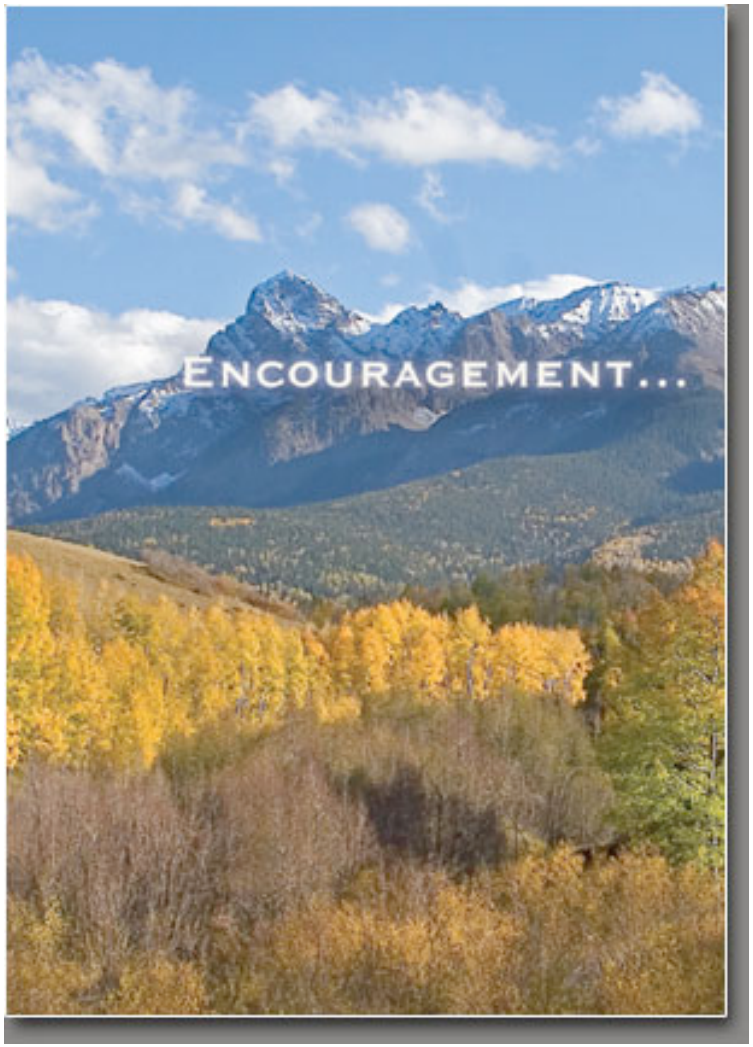
March 25, 2010

Problem

- Suppose you go to some trouble to build a system to satisfy the reference monitor concept
  - How would you evaluate that your system correctly satisfies that concept?

Practical Problem

- Commercial systems are not designed to satisfy the reference monitor concept
  - Can we do something to encourage improvement?
  - Can we layout a path that could result in a commercial system that satisfies the concept?



Assurance

PENNSTATE 1855

- A set of evaluations aiming to show that a system provides a correct security function
- Motivated by work on security kernels
  - *(1) Implement a specific security policy*
  - *(2) Design a verifiable protection behavior of the system as a whole*
  - *(3) Implementation must be shown to be faithful to the system's design*
- Develop distinct sets of requirements to be fulfilled for such an approach

Formal Assurance

- Throughout the 1970's and 1980's researchers examined methods to prove security properties
- Peter Neumann: *Provably Secure Operation System*
  - Formal design and proofs of security (1976)
- Padilla and Benzel: Evaluation of SCOMP
  - First "assured" system (1985)
- Popek, Kemmerer, Walker: Verification of UCLA UNIX Security Kernel (1980)
  - Completely Validated Software (ICSE 1989)
  - Integrating into Development Process (IEEE Soft 1990)

Formal Assurance

- Throughout the 1970's and1980's researchers examined methods to prove security properties
- Feiertag, Levitt, Robinson: Proving Multilevel Security of a System Design (SOSP 1977)
- Neumann, Feiertag, Levitt, Robinson: Software Development and Proofs of Multi-Level Security (ICSE 1976)

What Do We Want To Know?

PENNSTATE 1855

- Complete Mediation
  - Each structure member access to a security-sensitive object must be mediated (domination)
    - Find objects that enable an information flow between subjects
  - Mediation must authorize all dominated operations (AND of all accesses)
    - Can we compute all operations that are dominated?
  - Thus, uncommon accesses may require additional mediation (consistent across code)
    - Is a set of accesses embodied in an operation elsewhere? (mediated differently)

What Do We Want To Know?

- Tamperproofing
  - Start with known, good code and data (integrity verification)
  - Each information flow to kernel or TCB must be from trusted entity (Biba integrity) OR
  - Each information flow from an untrusted entity to kernel or TCB must be filtered (attack surface)
  - Filters must be acceptable (type safety, FSA, legal sequence of interfaces, …)

What Do We Want To Know?

- Verification
  - Code must correctly implement security function (security function: build queries, execute queries, responses)
  - Policy must correctly describe data security requirements – all authorized information flows (tamperproofing and user/app data)
- Design
  - Validate these in design
- Implementation
  - Verify mapping to implementation (what if no design?)

Assurance Criteria

- First proposal
  - Nibaldi proposed first criteria in 1979
  - "Laboratory Evaluations"
- Eventually led to Rainbow Series of TCSEC
- Criteria goals
  - Protection policy
  - Mechanisms contributing to effective enforcement of policy
  - Assurance that mechanisms are functioning

Rainbow Series

- Trusted Computer Systems Evaluation Criteria
- From 1983-1999
  - A variety of documents to help build secure systems
  - Password Management
  - Audit
  - Configuration Management
- Orange Book (1985)
  - Defined 6 classes of security systems
    - Function that the class provides
    - Requirements for verifying that implementation met the class
  - Requirements fall into a number of categories
    - Access control mechanism/policy
    - Authentication
    - Audit

Orange Book Classes

PENNSTATE 1855

- C1 and C2
  - Discretionary protection
    - Authentication, audit for discretionary access
    - Testing and documentation
  - C2 is the most common class for commercial products
- B1, B2, and B3
  - Labeled security protection:
    - Multi-level security (Bell-LaPadula)
    - More testing and more documentation
  - B1: MLS on some objects; B2: MLS on all
    - B2 also introduces covert channel protections and config mgmt
  - B3 more software engineering documentation
- A1: Verified protection
  - Requires correspondence between code and formal model

Common Criteria

- Started 1993 by US, Canada, and European Countries
- Attempt to identify a set of common criteria to evaluate information security
  - V1.0 1996, V2.0 1998, ISO Standard 15408 1999
  - A set of evaluation techniques used to vet technologies
  - … and tell which ones were good and bad (more or less).
  - This allows consumers of goods and services to know if the security advertised is as good as is claimed
    - Based on some specified evaluation criteria

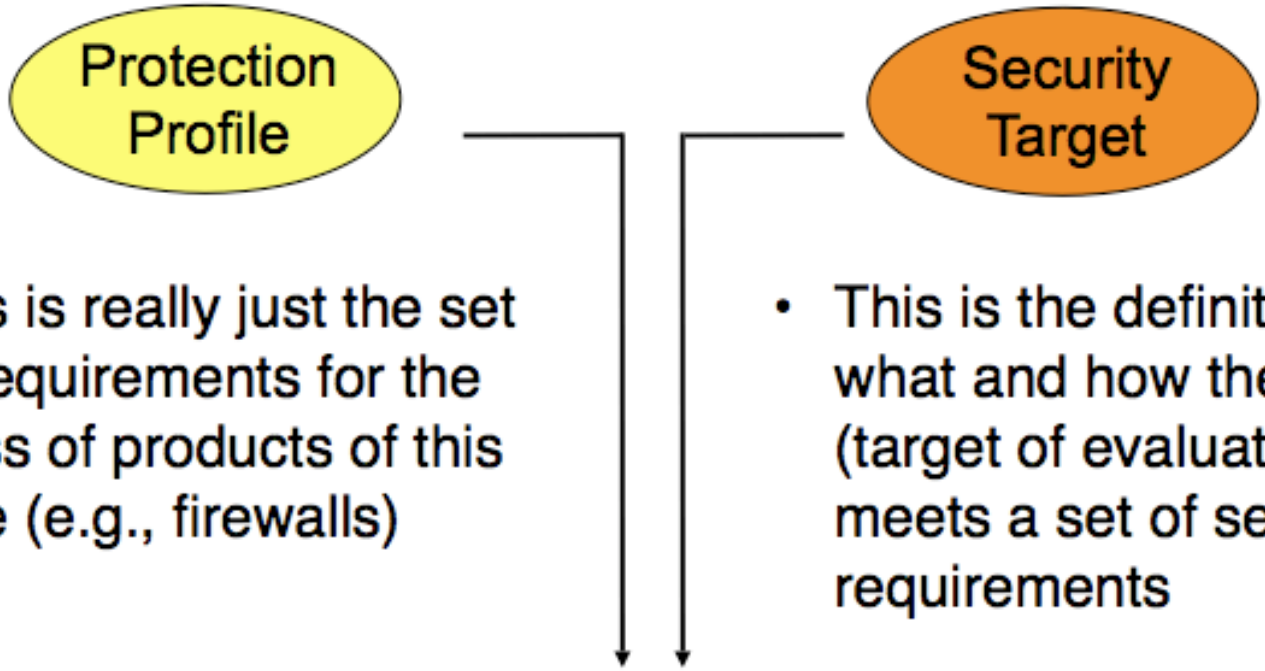
National Information Assurance Partnership

Common Criteria Certificate

Microsoft Corporation

Common Criteria

- Separate
  - Protection Profile
  - Assurance Level



- This is really just the set of requirements for the class of products of this type (e.g., firewalls)
- This is the definition of what and how the TOE (target of evaluation) meets a set of security requirements

EAL Levels

- EAL1: Functionally Tested
  - Breathing
- EAL2: Structurally Tested
  - High-level design
- EAL3: Methodically Tested and Checked
  - High-level design motivates testing
- EAL4: Methodically Designed, Tested, and Reviewed
  - Low-level design and vulnerability analysis
- EAL5: Semi-formally Designed and Tested
  - Rigorous development using (semi-)formal models
- EAL6: Semi-formally Verified Design and Tested
  - Low-level design
- EAL7: Formally Verified Design and Tested

Common Criteria in Practice

PENNSTATE 1855

- Linux is assured to:
  - EAL4 for Controlled Access Protection Profile
    - Discretionary access control with a low-level system design
- With LSM and SELinux (MLS)
  - EAL4 for Labeled Security Protection Profile
  - Done September 6, 2006
- Challenges
  - Upstream all code
    - Assure a mainline Linux kernel
  - Enable applications
    - E.g., Polymorphic file system
  - Package into distribution
    - That RedHat can deliver

Common Criteria in Practice

- Solaris 10 with Trusted Extensions (MLS)
  - Has been assured to EAL4+ for
  - LSPP, CAPP, and RBAC
  - Original Solaris was a C2 system, but is now EAL4 for CAPP and RBAC
- Windows Server 2003 and XP are EAL4 (for CAPP)
  - Windows 7 also aims for EAL4

Assurance Results

- Is Assurance achieving its goals?
  - Proving a system is "secure"
  - Encouraging the development of more secure systems
- Should we try something else?
  - What?
  - Can anything be automated?
  - On code?

seL4

Take Away

- VM Systems provide isolation
  - At OS granularity: some can be untrusted
- OS provides services used by applications
  - Access to devices demultiplexed among VMs
- Can we use VM isolation to prevent compromise of applications by OS compromise?
  - Proxos: use a "trusted" OS and redirect service requests
  - Overshadow: use OS as untrusted communication media